Date=20/08/2020
Lecture By=Shubham Joshi
Subject ⇒ Heaps-3

IN PREVIOUS LECTURE **(QUICK RECAP) Date-19/08/2020**	**In Today's Lecture (Overview)**
Heapify in Heap Types of Heaps Minimum Heap Maximum Heap Questions For Self Practice // Assignment for the Day	⇒ Insertion of heap Question=2(From leetcode) MCQs Questions For Self practice // CC For The Day

⇒ Insertion of heap

Problem Description

The program creates a binary max-heap and presents a menu to the user to perform various operations on it.

Problem Solution

1. Create a class BinaryHeap with an instance variable item set to an empty list. This empty list is used to store the binary heap.

2. Define methods size, parent, left, right, get, get_max, extract_max, max_heapify, swap and insert.

3. The method size returns the number of elements in the heap.

4. The method parent takes an index as argument and returns the index of the parent.

5. The method left takes an index as argument and returns the index of its left child.

6. The method right takes an index as argument and returns the index of its right child.

7. The method takes an index as argument and returns the key at the index.

8. The method get_max returns the maximum element in the heap by returning the first element in the list items.

9. The method extract_max returns the maximum element in the heap and removes it.

10. The method max_heapify takes an index as argument and modifies the heap structure at and below the node at this index to make it satisfy the heap property.

11. The method swap takes two indexes as arguments and swaps the corresponding elements in the heap.

12. The method insert takes a key as argument and adds that key to the heap.

Program/Source Code

```
class Heap:
    def __init__(self, heap_arr):
        self.heap_arr = heap_arr

    def pop_max_element(self):
        max_ele = self.heap_arr[0]
        self.heap_arr[0], self.heap_arr[len(heap_arr) - 1] =
self.heap_arr[len(heap_arr) - 1], self.heap_arr[0]
        self.heap_arr.pop()
        self.prune_down(0)
        return max_ele

    def prune_down(self, idx):
        if 2 * idx + 1 > len(self.heap_arr) - 1:
            return

        left = 2 * idx + 1
        right = 2 * idx + 2
        max_idx = idx


        if left <= len(self.heap_arr) - 1 and self.heap_arr[max_idx] <
self.heap_arr[left]:
            max_idx = left

        if right <= len(self.heap_arr) - 1 and self.heap_arr[max_idx] <
self.heap_arr[right]:
            max_idx = right

       if max_idx != idx:
            self.heap_arr[max_idx], self.heap_arr[idx] =
self.heap_arr[idx], self.heap_arr[max_idx]
            self.prune_down(max_idx)

    def build_heap(self):
        idx = len(self.heap_arr) - 1
        while idx >= 0:
            self.prune_down(idx)
            idx -= 1
        print("the heap build is ", self.heap_arr)

    def heap_sort(self):
        for i in range(len(self.heap_arr)):
            print(self.pop_max_element())


if __name__ == '__main__':
    heap_arr = [55, 33, 11, 22, 33, 77, 88, 100]
    heap = Heap(heap_arr)
    heap.build_heap()
    heap.heap_sort()
```

Question=2(From leetcode)

Find the kth largest element in an unsorted array

Note that it is the kth largest element in the sorted order, not the kth distinct element.

Example 1:

```
Input: [3,2,1,5,6,4] and k = 2
Output: 5
```

Example 2:

```
Input: [3,2,3,1,2,4,5,5,6] and k = 4
Output: 4
```

Note:

You may assume k is always valid, 1 ≤ k ≤ array's length.

Code

```
class Solution:
    def findKthLargest(self, nums, k):
        dict = {}
        for num in nums:
            if num not in dict:
                dict[num] = 1
            else:
                dict[num] += 1
        ans = []
        for i in range(min(nums), max(nums) + 1):
            if i in dict.keys():
                count = dict[i]
                while count:
```

```
                ans.append(i)
                count -= 1
                if len(ans) == len(nums) - (k - 1):
                    return ans[-1]
```

MCQs

1.**What is the time complexity for building a max / min heap ?**

A.O(n)

B.O(nlogn)

C.O(n2)

2.**What is the time complexity to get the max / min element from the heap ?**

A.O(k)

B.O(1)

C.O(logn)

D.O(nlogn)

3.**when will prune down not work for max heap ?**

A.when left and right are not max heap

B.when left and right are not min heap

4.**what is the best time complexity for finding the kth largest element in heap ?**

A.(n - k) logk

B.(n-k) logn n

C.(n-k)

D.nlogn

Questions For Self practice // CC For The Day

https://leetcode.com/problems/kth-largest-element-in-an-array/